LISTING OF CLAIMS

1. (currently amended) A method of creating a behavioral model to allow non-atomic behavioral simulation of process blocks in an electronic design, the method comprising:

receiving hardware design code describing a process block; ~~and~~

converting the hardware design code describing the process block to an assignment decision diagram (ADD) representation that is used by a simulator to simulate behavior of the process block; and

annotating the ADD representation of the process block with one or more control nodes **suitable for setting of a state within the process block** wherein when a particular control node is encountered, a state within the process block associated with the control node can be directly observed thereby ~~substantially~~ eliminating an atomic nature of the process block so as to provide a non-atomic analysis of the behavioral model.

2. (original) A method as recited in claim 1, wherein the assignment decision diagram representation comprises:

an assignment value portion representing the computation of values that are to be assigned to a storage unit of an output port, wherein the values are computed from current contents of storage units, input ports, or constants provided to the ADD;

an assignment condition portion connected as a data flow path representing the computation of a particular condition such that the end product of the condition computation is a binary value that evaluates to either *TRUE* or *FALSE*;

an assignment decision node (ADN) that selects a value from a set of input values computed by the assignment value portion based upon the conditions computed by the assignment condition portion; and

an assignment target portion that is provided with the selected value from the corresponding ADN corresponding to the true ADN condition.

3. (original) A method as recited in claim 1, wherein the control node is selected from the group comprising a query control node used to represent a conditional branch in a control flow, an evaluation/assignment control node used to represent an assignment operation, a null control node used as a place holder, and a suspend control node used to suspend execution of the process block.

4. (original) A method as recited in claim 3, wherein the suspend control node is selected from the group comprising an event type suspend control node used to suspend execution of the process block pending a pre-determined future event and a delay type suspend control node used to suspend execution of the process block for a for a specific length of time.

5. (original) A method as recited in claim 4, wherein the converting comprises:

synthesizing a circuit level parse tree based upon operational characteristics and schematic layout of the circuit being simulated contained within the hardware design code, wherein the parse tree includes a process token and a process block token hierarchically interconnected such that the process token branches from the process block token, the process token identifying a simulation process to be carried out within the process block;

traversing the parse tree and allocating a process block structure in the simulation object file when the process block token is encountered;

further traversing the parse tree to determine if the process token is available;

determining the type of simulation process identified by process token;

converting the identified simulation process to a corresponding assignment decision diagram; and

annotating the assignment decision diagram with a plurality of selected control nodes that are responsible for maintaining control flow through the simulator,

wherein each of the control nodes has a next pointer that is used by the simulator to point to a next process step in the simulation process, and wherein the control nodes are stored in a process block control node list contained within the object file.

6. (original) A method as recited in claim 1, wherein the hardware design code is selected from the group comprising Verilog and VHDL.

7. (currently amended) A behavioral model, provided on a machine readable medium, allowing non-atomic behavioral simulation of one or more process blocks in an electronic design, the model comprising an assignment decision diagram representation of the process block that can be used by a simulator to simulate behavior of the process block and including one or more control nodes **suitable for setting of a state within the process block** wherein when a particular control node is encountered, a state within the process block associated with the control node can be directly observed thereby ~~substantially~~ eliminating an atomic nature of the process block so as to provide the non-atomic behavioral simulation

8. (original) A behavioral model, provided on a machine readable medium, as recited in claim 7, wherein the assignment decision diagram representation comprises:

an assignment value portion representing the computation of values that are to be assigned to a storage unit of an output port, wherein the values are computed from current contents of storage units, input ports, or constants provided to the ADD;

an assignment condition portion connected as a data flow path representing the computation of a particular condition such that the end product of the condition computation is a binary value that evaluates to either *TRUE* or *FALSE*;

an assignment decision node (ADN) that selects a value from a set of input values computed by the assignment value portion based upon the conditions computed by the assignment condition portion; and

an assignment target portion that is provided with the selected value from the corresponding ADN corresponding to the true ADN condition.

9. (original) A behavioral model, provided on a machine readable medium, as recited in claim 7, wherein the control node is selected from the group comprising a query control node used to represent a conditional branch in a control flow, an evaluation/assignment control node used to represent an assignment operation, a null control node used as a place holder, and a suspend control node used to suspend execution of the process block.

10. (original) A behavioral model, provided on a machine readable medium, as recited in claim 9, wherein the suspend control node is selected from the group comprising an event type suspend control node used to suspend execution of the process block pending a pre-determined future event and a delay type suspend control node used to suspend execution of the process block for a for a specific length of time.

11. (currently amended) A method of compiling a simulation object file used by a simulator to simulate the operation of a digital circuit, wherein the simulation object file represents a behavioral model process block arranged to simulate the operation of the digital circuit by providing an output signal based upon an input signal, comprising:

(a) synthesizing a circuit level parse tree based upon operational characteristics and schematic layout of the circuit being simulated, wherein the parse tree includes a process token and a process block token hierarchically interconnected such that the process token branches from the process block token, the process token identifying a simulation process to be carried out within the process block;

(b) traversing the parse tree and allocating a process block structure in the simulation object file when the process block token is encountered;

(c) further traversing the parse tree to determine if the process token is available;

(d) determining the type of simulation process identified by process token;

(e) converting the identified simulation process to a corresponding assignment decision diagram; and

(f) annotating the assignment decision diagram (ADD) with a plurality of selected control nodes that are responsible for maintaining control flow through the simulator **and are suitable for setting of a state within the process block**, wherein when a particular control node is encountered, ~~a~~ **the** state within the process block associated with the control node can be directly observed thereby ~~substantially~~ eliminating an atomic nature of the process block,

wherein each of the control nodes has a next pointer that is used by the simulator to point to a next process step in the simulation process, and wherein the control nodes are stored in a process block control node list contained within the object file.

12. (original) A method as recited in claim 11, wherein the parse tree is formed of a plurality of sub-parse trees.

13. (original) A method as recited in claim 11, wherein the assignment decision diagram comprises:

an assignment value portion representing the computation of values that are to be assigned to a storage unit of an output port, wherein the values are computed from current contents of storage units, input ports, or constants provided to the ADD;

an assignment condition portion connected as a data flow path representing the computation of a particular condition such that the end product of the condition computation is a binary value that evaluates to either *TRUE* or *FALSE*;

an assignment decision node (ADN) that selects a value from a set of input values computed by the assignment value portion based upon the conditions computed by the assignment condition portion; and

an assignment target portion that is provided with the selected value from the corresponding ADN corresponding to the true ADN condition.

14. (original) A method as recited in claim 11, wherein the control node is selected from the group comprising a query control node used to represent a conditional branch in a control flow, an evaluation/assignment control node used to represent an assignment operation, a null control node used as a place holder, and a suspend control node used to suspend execution of the process block.

15. (original) A method as recited in claim 14, wherein the suspend control node comprises:

an event type suspend control node used to suspend execution of the process block pending a pre-determined future event.

16. (original) A method as recited in claim 14, wherein the suspend control node comprises:

a delay type suspend control node used to suspend execution of the process block for a for a specific length of time.

17. (original) A method as recited in claim 16, further comprising:

(g) setting a next pointer of a last control node to a first control node; and

(h) repeating (c) – (g) until no process tokens are available.

18. (original) A method as recited in claim 17, wherein when it is determined that the process token identifies a process conditional, the converting the process further comprises:

allocating a query control node to the process block's control node list;

recursively converting a true sub-parse tree to a corresponding true conditional ADD and a plurality of corresponding true eval CNs, wherein the true conditional ADD includes a true conditional ADN; and

recursively converting a false sub-parse tree to a corresponding false conditional ADD and a corresponding false eval CN.

19. (original) A method as recited in claim18, wherein when it is determined that the process token identifies a process conditional, the annotating the ADD further comprises:

mapping the query CN to the true conditional ADN;

setting each of the conditional inputs of the true ADN to appropriate ones of the true eval and false eval CNs;

allocating a null CN to the process block CN list;

setting a CN pointer associated with a last one of the plurality of true eval CNs and a CN pointer associated with a last one of the plurality of false eval CNs to the null CN; and

identifying the query CN as a first CN and the null CN as a last CN.

20. (original) A method as recited in claim 17, wherein when it is determined that the type of token is a process loop type token, the loop process having a loop entry condition and a loop exit condition suitable for entering and exiting, respectively, a corresponding loop expression and a loop body, the converting the process further comprises:

allocating a query CN to the process block CN list;

recursively converting a loop expression sub-parse tree to a corresponding loop expression ADD; and

recursively converting a loop body sub-parse tree to a loop body ADD.

21. (original) A method as recited in claim 20, wherein when it is determined that the process token identifies a process loop, the annotating the ADD further comprises:

annotating the loop expression ADD with appropriate loop expression control nodes;

mapping the query CN to the loop expression ADD;

annotating the loop body ADD with appropriate loop body control nodes;

returning a loop body first CN and a loop body last CN;

allocating a null CN;

pointing the loop body last CN to the null CN;

pointing the loop entry condition to the loop body first CN;

pointing the loop exit condition to the null CN; and

identifying the query CN as a first CN and the null CN as a last CN.

22. (original) A method as recited in claim 17, wherein when it is determined that the type of token is a process suspend type token, the converting the process further comprises:

recursively converting a suspend sub-parse tree to a corresponding suspend ADD

23. (original) A method as recited in claim 22, wherein when it is determined that the type of token is a process suspend type token, the annotating the ADD further comprises:

determining the process suspend type;

if it is determined that the suspend type is a delay suspend type, then allocating a delay suspend CN to the process block CN list;

if it is determined that the suspend type is an event suspend type, then allocating an event suspend CN to the process block CN list;

setting the allocated control node's reason to suspend pointer to the suspend ADD; and

identifying the allocated CN as a first CN and a last CN.

24. (original) A method as recited in claim 17, wherein when it is determined that the process is a process assignment, the converting the process further comprises:

recursively converting the assignment sub-parse tree to a corresponding assignment ADD.

25. (original) A method as recited in claim 24, wherein when it is determined that the type of token is a process assignment type token, the annotating the assignment ADD further comprises:

allocating an assignment CN;

storing the allocated CN in the process block CN list;

mapping the CN to the assignment ADD; and

identifying the allocated CN as the first CN and the last CN.

26. (original)A method as recited in claim 11, wherein the digital circuit being simulated is a programmable logic device.

27. (currently amended) A computer program project comprising computer program instructions provided on a computer readable medium, the computer program instructions specifying a method of compiling a simulation object file used by a simulator to simulate the operation of a digital circuit, wherein the simulation object file represents a behavioral model process block arranged to simulate the operation of the digital circuit by providing an output signal based upon an input signal, the method comprising:

synthesizing a circuit level parse tree based upon operational characteristics and schematic layout of the circuit being simulated, wherein the parse tree includes a process token and a process block token hierarchically interconnected such that the process token branches from the process block token, the process token identifying a simulation process to be carried out within the process block;

traversing the parse tree and allocating a process block structure in the simulation object file when the process block token is encountered;

further traversing the parse tree to determine if the process token is available;

determining the type of simulation process identified by process token;

converting the identified simulation process to a corresponding assignment decision diagram; and

annotating the assignment decision diagram (ADD) with a plurality of selected control nodes **suitable for setting of a state within the process block** wherein when a particular control node is encountered, a state within the process block associated with the control node can be directly observed thereby ~~substantially~~ eliminating an atomic nature of the process block, wherein each of the control nodes has a next pointer that is used by the simulator to point to a next process step in the simulation process, and wherein the control nodes are stored in a process block control node list contained within the object file.

28. (currently amended) An apparatus for creating a behavioral model to allow non-atomic behavioral simulation of process blocks in an electronic design, comprising:

a schematic editor for providing hardware design code that describes a process block;

a synthesizer coupled to the schematic editor for converting the hardware design code to an assignment decision diagram (ADD) representation;

an annotator coupled to the synthesizer for annotating the assignment decision diagram with one or more control nodes, and

a simulator for simulating the electronic design using the annotated assignment decision diagram, wherein the one or more control nodes are wherein when a particular control node is encountered, a state within the process block associated with the control node can be directly observed thereby substantially eliminating an atomic nature of the process block.